תכנות באינטרנט

ADO.Net

Data Reader

Params

גלעד מרקמן

קריית החינוך פארק המדע,
נס ציונה

Microsoft

ASP.NET Core

DataReader

* רשות

קריאת נתונים מטבלה בשיטה המקושרת

- בשיטה המקושרת אנחנו קוראים נתונים מטבלה באמצעות iterator העובר על כל שורה בטבלה, כשהחיבור לבסיס הנתונים פתוח.
- הפעולה ExecuteReader מחזירה לנו iterator כזה.
- ה iterator "צועד" רק קדימה.
- הפעולות שניתן לבצע הן:

reader.HasRows	אמת אם הוחזרו שורות
reader.Read()	קרא את השורה הבאה מבסיס הנתונים. גם לפני קריאה ראשונה
Reader.getValue(0)	מחזיר אובייקט של העמודה הראשונה – מחייב המרה
reader.GetString(1)	מחזיר מחרוזת - יש לוודא שסוג הנתונים תואם
reader.GetBoolean(5)	מחזיר ערך בוליאני
Reader.GetName(2)	מחזיר שם העמודה

- בסיום יש לסגור את ה iterator.

הפעולה Helper.GetDataREader

- הפעולה מקבלת מחרוזת עם פקודה SQL – פקודת SELECT.
- הפעולה מחזירה אובייקט iterator אשר מאפשר לשלוף שורה שורה מהטבלה שהתקבלה על ידי פקודת SQL.
- הפרמטר CommandBehavior.CloseConnection מבטיח שסגירת ה reader יסגור את החיבור.

```
public SqlDataReader GetDataReader(string SQL)
{
    // התחברות למסד הנתונים
    SqlConnection con = new SqlConnection(conString);

    // בניית פקודת SQL
    SqlCommand cmd = new SqlCommand(SQL, con);

    con.Open();
    // Command behavior insure closing the reader will close the connection
    SqlDataReader reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);
    return reader;
}
```

הקוד בצד השרת

- נגדיר במודל של צד השרת מאפיין Reader.

- בפעולה OnGet ניצור Iterator לטבלה ונשמור אותו במאפיין.

```
public SqlDataReader Reader { get; set; }

public void OnGet()
{
    Helper helper = new Helper();
    string SQL = "SELECT * FROM Users";
    Reader = helper.GetDataReader(SQL);
}
```

Params
SQL Injection

* רשות

יצירת הטבלה

```
<table class='table'>
    <thead>
        <tr>
            @if (Model.Reader.HasRows)
            {
                for (int i = 0; i < Model.Reader.FieldCount; i++)
                {
                    <th>@Model.Reader.GetName(i)</th>
                }
            }
        </tr>
    </thead>
    <tbody>
        @while (Model.Reader.Read())
        {
            <tr>
                @for (int i = 0; i < Model.Reader.FieldCount; i++)
                {
                    <td>@Model.Reader.GetValue(i)</td>
                }
            </tr>
        }
    </tbody>

</table>
@{
    Model.Reader.Close();
}
```

- ניצור את הטבלה באמצעות לולאות דומות, אך במקום להשתמש ב DataTable נשתמש ב Iterator.

- נשים לב, כי בסיום יצירת הטבלה יש לסגור את ה reader.

כתיבת command באמצעות פרמטרים

```
1 reference
public int ParamExecuteNonQuery()
{
    SqlConnection con = new SqlConnection(_connectionString);

    String SQL = "UPDATE Users " +
        "Set LastName = @lastName " +
        "WHERE LastName LIKE @filter";

    SqlCommand cmd = new SqlCommand(SQL, con);

    SqlParameter param = new SqlParameter ();
    param.ParameterName = "lastName";
    param.Value = "Rom";
    param.SqlDbType = SqlDbType.NVarChar;
    cmd.Parameters.Add(param);

    param = new SqlParameter ();
    param.ParameterName = "filter";
    param.Value = "Yaakov";
    param.SqlDbType = SqlDbType.Char;
    cmd.Parameters.Add(param);

    con.Open();
    int n = cmd.ExecuteNonQuery();
    return n;
}
```

- ניתן לכתוב את פקודת ה SQL המשמשת אותנו לבניית האובייקט SqlCommand באמצעות פרמטרים.
- דרך זו מפחיתה את שגיאות ההקלדה.
- כמו כן, שיטה זו מונעת פרצת אבטחה המכונה SQL Injection.

סיכום

- השלבים בעבודה בשיטת עבודה מקושרת.
- הפקודות בשיטה המקושרת:
  - Command.ExecuteScalar
  - Command.ExecuteNonQuery
  - Command.ExecuteReader
- שימוש באובייקט DataReader
  - reader.HasRows;
  - reader.Read();
  - Reader.getValue(0);
  - reader.GetString(1);
  - reader.GetBoolean(5);
  - Reader.GetName(2);